10/521963

R E P U B L I Q U E F R A N Ç A I S E

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

PCT/FR03/02245

REC'D 20 OCT 2003
WIPO PCT

BREVET D'INVENTION

CERTIFICAT D'UTILITÉ - CERTIFICAT D'ADDITION

COPIE OFFICIELLE

Le Directeur général de l'Institut national de la propriété industrielle certifie que le document ci-annexé est la copie certifiée conforme d'une demande de titre de propriété industrielle déposée à l'Institut.

Fait à Paris, le 06 MAI 2003

Pour le Directeur général de l'Institut
national de la propriété industrielle
Le Chef du Département des brevets

Martine PLANCHE

DOCUMENT DE PRIORITÉ

PRÉSENTÉ OU TRANSMIS
CONFORMÉMENT À LA
RÈGLE 17.1.a) OU b)

INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE
SIEGE
26 bis, rue de Saint Petersbourg
75800 PARIS cedex 08
Téléphone : 33 (0)1 53 04 53 04
Télécopie : 33 (0)1 53 04 45 23
www.inpi.fr

141102

BEST AVAILABLE COPY

INPI
INSTITUT NATIONAL DE LA PROPRIÉTÉ INDUSTRIELLE

26 bis, rue de Saint Pétersbourg
75800 Paris Cedex 08
Téléphone : 01 53 04 53 04 Télécopie : 01 42 94 86 54

BREVET D'INVENTION
CERTIFICAT D'UTILITÉ

Code de la propriété intellectuelle - Livre VI

N° 11354*01

REQUÊTE EN DÉLIVRANCE 1/2

Important ! Remplir impérativement la 2ème page.

Cet imprimé est à remplir lisiblement à l'encre noire — DB 540 W / 190600

Réservé à l'INPI	
REMISE DES PIÈCES DATE 22 JUIL 2002 LIEU 75 INPI PARIS N° D'ENREGISTREMENT NATIONAL ATTRIBUÉ PAR L'INPI 0209287 DATE DE DÉPÔT ATTRIBUÉE PAR L'INPI 22 JUIL. 2002	1 NOM ET ADRESSE DU DEMANDEUR OU DU MANDATAIRE À QUI LA CORRESPONDANCE DOIT ÊTRE ADRESSÉE CABINET NETTER 36 avenue Hoche 75008 PARIS
Vos références pour ce dossier *(facultatif)* INRIA Aff. 59 (120777)	

Confirmation d'un dépôt par télécopie ☐ N° attribué par l'INPI à la télécopie

2 NATURE DE LA DEMANDE	Cochez l'une des 4 cases suivantes
Demande de brevet	☒
Demande de certificat d'utilité	☐
Demande divisionnaire	☐
Demande de brevet initiale	N° Date / /
ou demande de certificat d'utilité initiale	N° Date / /
Transformation d'une demande de brevet européen *Demande de brevet initiale*	☐ N° Date / /

3 TITRE DE L'INVENTION (200 caractères ou espaces maximum)

Perfectionnement à la compression de données numériques.

4 DÉCLARATION DE PRIORITÉ OU REQUÊTE DU BÉNÉFICE DE LA DATE DE DÉPÔT D'UNE DEMANDE ANTÉRIEURE FRANÇAISE	
	Pays ou organisation Date / / N°
	Pays ou organisation Date / / N°
	Pays ou organisation Date / / N°
	☐ **S'il y a d'autres priorités, cochez la case et utilisez l'imprimé «Suite»**

5 DEMANDEUR		☐ **S'il y a d'autres demandeurs, cochez la case et utilisez l'imprimé «Suite»**
Nom ou dénomination sociale		INRIA INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE
Prénoms		
Forme juridique		Etablissement Public national à caractère scientifique et technologique
N° SIREN		
Code APE-NAF		
Adresse	Rue	Domaine de Voluceau - Rocquencourt - BP 105
	Code postal et ville	78153 LE CHESNAY CEDEX
Pays		France
Nationalité		française
N° de téléphone *(facultatif)*		
N° de télécopie *(facultatif)*		
Adresse électronique *(facultatif)*		

INPI
INSTITUT NATIONAL DE LA PROPRIÉTÉ INDUSTRIELLE

BREVET D'INVENTION
CERTIFICAT D'UTILITÉ

REQUÊTE EN DÉLIVRANCE 2/2

Réservé à l'INPI

REMISE DES PIÈCES
DATE 22 JUIL 2002
LIEU 75 INPI PARIS
N° D'ENREGISTREMENT NATIONAL ATTRIBUÉ PAR L'INPI 0209287

DB 540 W / 190600

Vos références pour ce dossier : *(facultatif)*		INRIA Aff. 59 (120777)
6 MANDATAIRE		
Nom		PLAÇAIS
Prénom		Jean-Yves
Cabinet ou Société		Cabinet NETTER
N °de pouvoir permanent et/ou de lien contractuel		
Adresse	Rue	36 avenue Hoche
	Code postal et ville	75008 PARIS
N° de téléphone *(facultatif)*		01 58 36 44 22
N° de télécopie *(facultatif)*		01 42 25 00 45
Adresse électronique *(facultatif)*		
7 INVENTEUR (S)		
Les inventeurs sont les demandeurs		☐ Oui ☒ Non **Dans ce cas fournir une désignation d'inventeur(s) séparée**
8 RAPPORT DE RECHERCHE		**Uniquement pour une demande de brevet (y compris division et transformation)**
Établissement immédiat ou établissement différé		☒ ☐
Paiement échelonné de la redevance		**Paiement en deux versements, uniquement pour les personnes physiques** ☐ Oui ☐ Non
9 RÉDUCTION DU TAUX DES REDEVANCES		**Uniquement pour les personnes physiques** ☐ Requise pour la première fois pour cette invention *(joindre un avis de non-imposition)* ☐ Requise antérieurement à ce dépôt *(joindre une copie de la décision d'admission pour cette invention ou indiquer sa référence)* :
Si vous avez utilisé l'imprimé «Suite», indiquez le nombre de pages jointes		

10 SIGNATURE DU DEMANDEUR OU DU MANDATAIRE (Nom et qualité du signataire)

N° Conseil 92-1197 (B) (M)
Jean-Yves PLAÇAIS

VISA DE LA PRÉFECTURE OU DE L'INPI

L. MARIELLO

La loi n°78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés s'applique aux réponses faites à ce formulaire.
Elle garantit un droit d'accès et de rectification pour les données vous concernant auprès de l'INPI.

Perfectionnement à la compression de données numériques

L'invention concerne la compression de données numériques, notamment pour des signaux multimédia.

Pour réduire le débit de transmission des données numériques, on les comprime, en cherchant à s'approcher du maximum théorique que les spécialistes appellent "l'entropie du signal". On utilise souvent des codes statistiques aussi appelés codes à longueurs variables, par exemple les codes de Huffman.

Encore faut-il se préoccuper également des effets des perturbations du signal. En effet, celles-ci réduisent le débit utile : lorsque des codes correcteurs d'erreurs détectent une erreur à la réception, il faut re-transmettre la partie correspondante du signal.

Les solutions actuelles, sur lesquelles on reviendra, se fondent sur l'hypothèse qu'une certaine qualité de service du transport des données est garantie. Ceci permet d'atteindre un taux d'erreur résiduel quasi nul à la réception, après décodage. Mais cette hypothèse de taux d'erreur résiduel quasi nul n'est plus vraie lorsque les caractéristiques des canaux varient dans le temps (canaux non stationnaires), notamment dans les réseaux sans fil et mobiles.

Le besoin se fait donc sentir de solutions qui soient moins dépendantes du caractère stationnaire des canaux de transmission.

La présente invention vient proposer des avancées en ce sens.

Selon un aspect de l'invention, il est proposé un codeur de compression de données numériques, qui comprend:

- une entrée (physique ou non) pour un premier flux de données (S_H), et un second flux de données (S_L),
- une table de codage, contenant une correspondance entre des symboles du premier flux de données et des mots de code, où, pour certains symboles, il existe plusieurs mots, dits redondants, correspondant au même symbole, et

- un module de traitement pour coder les symboles du premier flux de données d'après la table, en choisissant parmi les mots redondants, en fonction d'une partie au moins du second flux de données.

Selon différents autres aspects:

- les mots de code peuvent être de longueur fixe,
- le module de traitement comprend:
  . une fonction de calcul de la capacité de multiplexage courante du premier flux de données (S_H), au vu de la table de codage, et
  . une fonction d'extraction, dans le second flux de données (S_L), d'une partie multiplexée, déterminée d'après la capacité de multiplexage courante, pour être portée par lesdits mots redondants.
- le codeur comprend une transformation d'un flux binaire en un flux de variable multi-valuée, en particulier en utilisant les transformations décrites dans la table 2 ci-après.
- le second flux de données est préalablement encodé.
- le reste du second flux de données est concaténé aux données transmises.

L'invention vise également un décodeur, propre à effectuer les opérations inverses ou réciproques de celle du codeur, dans ses différents aspects.

L'invention vise encore un procédé de compression de données numériques, qui comprend les étapes suivantes:

a. établir une table de codage, contenant une correspondance entre des symboles du premier flux de données et des mots de code, où, pour certains symboles, il existe plusieurs mots, dits redondants, correspondant au même symbole, et
b. coder les symboles d'un premier flux de données d'après la table, en choisissant parmi les mots redondants, en fonction d'une partie au moins d'un second flux de données.

Ce procédé peut intégrer les autres aspects du codage.

Enfin, l'invention vise également le procédé de décompression de données numériques, comprenant les étapes réciproques de celles du procédé de compression.

D'autres caractéristiques et avantages de l'invention apparaîtront à l'examen de la description détaillée ci-après, et des dessins annexés, sur lesquels :

- la figure 1 illustre de façon schématique un procédé de création de codes,
- la figure 2 est un diagramme formant vue générale du processus d'encodage,
- la figure 2A ou table 1 illustre un exemple simplifié de code multiplexé, à quatre éléments,
- la figure 3 illustre un premier mode de réalisation d'un processus détaillé d'encodage,
- la figure 4 illustre une variante de réalisation du processus de la figure 3,
- la figure 5 illustre un exemple de création d'une capacité de stockage par l'attribution de plusieurs mots de code à un symbole, un flux de données q pouvant être conjointement stocké, et
- la figure 6 ou table 2 illustre des transformations utilisées un exemple où le paramètre f_v est égal à 5.

En outre:

- l'annexe 1 contient des expressions utilisées dans la présente description, et
- l'annexe 2 contient des algorithmes en langage naturel utilisés dans la présente description.

Les dessins et les annexes à la description comprennent, pour l'essentiel, des éléments de caractère certain. Ils pourront donc non seulement servir à mieux faire comprendre la description, mais aussi contribuer à la définition de l'invention, le cas échéant.

La description détaillée ci-après est structurée en sections, qui facilitent la lecture et les renvois.

1. Contexte

Tout système de compression de signaux multimédia (image, vidéo, audio, parole) fait appel à des codes statistiques aussi appelés codes à longueurs variables. Ceux-ci permettent d'obtenir des débits approchant ce que les spécialistes appellent "l'entropie du signal". Les codes les plus utilisés dans les systèmes existants (en particulier dans les standards) sont les codes de Huffman.

Plus récemment, on a vu un regain d'intérêt pour les codes arithmétiques en raison de leurs performances accrues en terme de compression. Ils permettent en effet de découpler le processus d'encodage du modèle supposé de la source. Ceci permet aisément d'utiliser des modèles statistiques d'ordre supérieur.

Jusque récemment, la conception des systèmes de compression se faisait en supposant une qualité de service transport garantie. On supposait en effet que les couches inférieures du modèle OSI incorporent des codes correcteurs d'erreurs garantissant un taux d'erreur résiduel vu de l'application quasi-nul.

Les codes à longueurs variables pouvaient donc être largement utilisés malgré leur forte sensibilité au bruit de transmission. Toute erreur dans le train binaire peut engendrer une dé-synchronisation du décodeur et donc une propagation des erreurs sur la suite des informations décodées.

Pour pallier ce problème de propagation, les standards de premières générations (H.261, H.263, MPEG-1, MPEG-2) ont incorporé dans la syntaxe du train binaire transmis des marqueurs de synchronisation. Ce sont des mots de code longs (16 ou 22 bits constitués d'une suite de 15 ou 21 bits à '1' suivis d'un '0') non émulables par des erreurs se produisant sur les autres mots du codes et qui peuvent donc être reconnus par le décodeur avec une probabilité proche de '1'.

Cela conduit à structurer le train binaire en paquets délimités par ces marqueurs de synchronisation. Cela permet de confiner la propagation des erreurs au sein du paquet. Cependant si une erreur intervient en début de paquet la suite du paquet peut être perdue. En outre, la périodicité de ces marqueurs de synchronisation doit être restreinte pour éviter une perte trop grande en efficacité de compression.

Cette hypothèse de taux d'erreur résiduel quasi nul n'est plus vraie dans les réseaux sans fil et mobiles, dont les caractéristiques des canaux varient dans le temps (canaux non stationnaires). Ce taux d'erreur résiduel vu par le décodeur des signaux de source est souvent loin d'être négligeable.

Les nouveaux standards (H.263+ et MPEG-4) ont alors fait appel à des codes à longueurs variables réversibles (RVLC []). La particularité de ces codes est qu'ils peuvent être décodés du premier vers le dernier bit d'un paquet, et, à l'inverse, du dernier vers le premier bit du paquet.

Si une erreur s'est produite en milieu de paquet, cette symétrie du code permet de confiner la propagation des erreurs sur un segment en milieu de paquet au lieu de la propager jusqu'à la fin du paquet délimité par un marqueur de synchronisation. Cependant, la symétrie du code engendre une perte en efficacité de compression par rapport à un code de Huffman de l'ordre de 10 %. En outre les codes à longueurs variables réversibles n'évitent pas complètement le problème de propagation des erreurs : si une erreur se produit en début et en fin de paquet, tout le paquet risque d'être erroné.

La conception de codes qui soient à la fois performants en compression (i.e., qui permettent d'approcher l'entropie de la source), tout en étant robustes au bruit de transmission, constitue donc un enjeu important, notamment pour les futurs systèmes de communication multimédia (image, vidéo, audio, parole) mobiles. Pour ces systèmes de nouveaux standards sont à l'étude à la fois au sein de l'ITU (International Telecommunication Union) et de l'ISO (International Standard Organization).

Bien que les standards occupent une place prépondérante dans le secteur des télécommunications, une telle famille de codes peut aussi trouver des applications sur des marchés de niche faisant appel à des solutions propriétaires.

2 Description

2.1 Principe général

Le principe général du procédé consiste à créer des codes de longueur fixée pour les données de source importantes (flux s_H), en attribuant plusieurs mots de codes à chaque réalisation possible de cette source.

Ainsi, pour transmettre un symbole, il est possible de choisir parmi les différentes représentations possibles de celui-ci. Ce choix, qui est une variable multi-valuée, définit une capacité de stockage qui va pouvoir être utilisée pour transmettre conjointement d'autres données (cf. exemple de la figure 5). Ce sont les données d'importance moindre, représentées par un flux noté s_L, qui vont être représentées via la représentation multiple des symboles.

Ce document présente le procédé de création de ces codes, et le procédé de codage. Une variante permettant d'éviter les calculs sur entiers longs est également présentée. Le processus de décodage associé est effectué en procédant aux opérations inverses de celles de l'encodage.

2.2 Création des codes multiplexés

La source importante, s_H , prend ses valeurs dans un alphabet à Ω éléments, qui peut être défini par l'expression (E1) en annexe.

La loi μ de probabilité d'apparition des symboles sur cet alphabet est supposée connue. On note μ_i la probabilité associée au symbole a_i de l'alphabet de la source s_H , comme représenté par l'expression (E2).

Le procédé de création des codes, représenté sur la figure 1, peut se décomposer en 2 étapes :

- Pour chaque symbole a_i , choix du nombre N_i de mots de code affectés à ce symbole,
- Allocation des mots de codes aux symboles.

2.2.1 Sélection des paramètres de codes c et (N_i)

1. Choix d'un paramètre c de longueur de mot de code, en nombre de bits. Usuellement c = 14. Ceci définit 2 c mots de codes, à répartir entre les symboles de l'alphabet A.

2. On partitionne l'ensemble des symboles de l'alphabet A en deux sous-ensembles A_m et A_M . Le premier est l'ensemble des symboles a_i dont la probabilité μ_i est inférieure ou égale à 1 / 2 c , le second est son complémentaire dans A. Les cardinaux de ces ensembles sont notés respectivement Ω_m et Ω_M .

3. La loi de probabilité $\tilde{\mu}$ sur les symboles de A_M est alors calculée. Elle est donnée par l'expression (E3).

4 Le nombre de mots de codes par symbole est alors choisi de manière à vérifier approximativement l'expression (E4), sous la contrainte de l'expression (E5) . Dans ce but, un algorithme classique d'optimisation peut être utilisé.

Variante : Dans une variante, Une étape additionnelle est ajoutée (5var), et les étapes 1 et 4 se déclinent respectivement sous la forme suivante :

- 1var - Soient $f_1 = 2$, $f_2 = 3$, ..., f_ν les ν premiers nombres premiers. En plus du paramètre c, un nombre premier f_ν est également choisi. Usuellement $f_\nu = 5$.

- 4var On procède de la même manière, mais en ajoutant un contrainte supplémentaire pour le choix du nombre de mots de code N_i associés à chaque symbole : la décomposition en facteurs premiers de tous les N_i ne doit pas contenir de facteur premier supérieur à f_ν .

- 5var La décomposition en facteurs premiers de chaque N_i est alors effectuée, et on calcule pour tout N_i le nombre de fois que chaque facteur premier f_j , avec $1 \leq j \leq \nu$, apparaît dans cette décomposition. Ce nombre est noté $\alpha_{i,j}$, où i indexe le symbole a_i considéré et j indexe le nombre premier f_j considéré.

2.2.2 Allocation des mots de codes aux symboles

L'ordre lexicographique de l'étiquetage binaire (0000, 0001, . . .) est utilisé pour affecter aux différents symboles le nombre N_i de mots de codes déterminé à l'étape précédente.

L'ensemble des mots de codes ainsi associés à un symbole a_i est appelé classe d'équivalence de a_i , et notée C_i .

On associe alors aux mots de codes de chaque classe d'équivalence une valeur entre 0 et N_i -1, appelée ici état. Cette valeur identifie le mot de code au sein de la classe d'équivalence.

Ainsi, à chaque mot de code c_i est associé un symbole a_i et une variable d'état comprise entre 0 et N_i - 1, comme illustré par l'expression (E6).

Un exemple de code ainsi construit est donné dans la table 1.

2.3 Processus d'encodage

Le processus d'encodage (cf. figure 2 et 3) se décompose de la manière suivante :

1. Le flux de données d'importance moindre subit est encodé en une suite binaire
   $b = (b_1 , b_2, ..., b_{KB})$.
   A cet effet, un codeur réversible de type Huffman ou codeur arithmétique (non restrictif) peut être utilisé. Cela aboutit à la génération d'une suite de bits, notée b.

2. De la séquence de symboles $s_1 , s_2 , ... , s_{KH}$ du flux s_H, on déduit les valeurs $n_1 , n_2 ,... , n_{KH}$ associées.

3. On en déduit la valeur Λ, ici selon l'expression (E7). On calcule le nombre K'_B de bits qui vont pouvoir être stockés en utilisant la redondance intrinsèque des codes multiplexés, selon l'expression (E8)

4. Les K'_B derniers bits du flux b sont utilisés pour calculer un entier long γ, donné ici par la relation (E9)

α

5. La valeur γ permet alors de calculer les états q_t , $1 \leq t \leq K_H$, en utilisant par exemple une méthode de décomposition euclidienne généralisée, comme illustré dans l'algorithme (A1) annexé.

6. Pour tout t tel que $1 \leq t \leq K_H$, la connaissance du symbole s_t et de l'état q_t calculé à l'étape précédente permet de choisir le mot de code dans la table des mots de codes multiplexés.

7. Les K_H - K'_H bits du flux d'importance moindre sont alors concaténés à la séquence de mots de codes multiplexés précédemment évaluée.

2.4 Variante au processus d'encodage

La variante du processus d'encodage (cf. figure 4) se décompose de la manière suivante :

1. Cf. étape 1 de la section 2.3

2. Cf. étape 2 de la section 2.3

3. Le nombre total de fois où chaque facteur premier f_j apparaît dans l'ensemble des décompositions en facteurs de la séquence n_t est alors calculé. Il est noté d_j dans la suite, et représente le nombre de variables f_j-valuées qui peuvent être multiplexées avec le flux s_H .

4. On choisit alors les transformations qui vont être utilisées pour transformer le train binaire en ces variables f_j-valuées. Ces transformations dépendent de la valeur de f_v choisie. Les transformations utilisés pour $f_v = 5$ sont présentées dans la table 2.

Elles se présentent sous la forme illustrée dans les expressions (E10) annexées.

Ainsi elles prennent u_T bits en entrée et les transforment en respectivement $v_{T,1}$, $v_{T,2}$,..., $v_{T,v}$ variables 2, 3, . . . , f_v -valuées. Or le nombre requis de variables de chaque type est connu: pour chaque type de variable f_j , il est d_j (cf. étape 3).

L'algorithme A2 annexé peut être utilisé pour calculer le nombre g_{Tz} de fois que la transformation T z doit être utilisée.(Les transformations sont supposées être classées dans l'ordre décroissant de leur pertinence dans la table) .

5. Ayant choisi le nombre de transformations de chaque type qui seront utilisées, on les applique à la fin au flux binaire b.

- Les u_T bits en entrée sont vus comme la représentation binaire d'un entier e.

- Cet entier est alors décomposé en plusieurs variables f_j -valuées, comme précisé dans les expressions (E10). On note $e_{r,j}$ ces variables, où :
  j indique que la valeur obtenue est la réalisation d'une variable f_j-valuée, et
  r indique le numéro de variable f_j -valuée.

Les valeurs des $e_{r,j}$ peuvent s'obtenir à partir de e avec le procédé de l'algorithme A3.

A l'issue de cette étape, v séquences de variables sont disponibles :

le premier, noté F_1, est une séquence de longueur d_1 de variables 2-valuée (bits),

le j-ème, noté F_j , est une séquence de longueur d_j de variables f_j-valuée. Des pointeurs de positions sont associés aux séquences, ils sont initialement positionnés sur le début de chaque séquence.

6. A partir de ces variables, on calcule le flux d'état
$q = (q_1, q_2 ,..., q_{KH})$.

Pour tout t tel que $1 \leq t \leq K_H$, la décomposition en facteurs premiers de n_t permet de déterminer le nombre de variables de chaque type (2-valuées,. . .,f_j-valuées,. . .,f_v-valuées) qui permettront de créer la variable n_t-valuée (q_t). Notons qu'à la fin de cette étape, toutes les variables des flux F_j ont été utilisées.

α

7. Pour tout t tel que $1 \leq t \leq KH$, la connaissance du symbole s_t et de l'état q_t calculé à l'étape précédente permet de choisir le mot de code dans la table des mots de codes multiplexés.

8. Les $K_H - K'_H$ bits du flux d'importance moindre sont alors concaténés à la séquence de mots de codes multiplexés précédemment évaluée.

Ainsi, l'invention permet le multiplexage de deux flux de données S_H et S_L, afin de diminuer la sensibilité aux erreurs de l'un d'eux S_H, désigné comme plus important ou prioritaire. Ces deux flux peuvent être distingués dans la même source de signaux, notamment comme dans les quelques exemples suivants de sources S_H et S_L :

- basses fréquences et hautes fréquences extraites par décomposition multi-résolution (par bancs de filtres, transformées en ondelettes) d'un signal,

- information de texture (ex : coefficients DCT, coefficients ondelettes) et information de mouvement,

- bits de poids forts et bits de poids faibles des coefficients ondelettes ou des échantillons quantifiés d'un signal.

Bien entendu, l'énumération ci-dessus n'a aucun caractère exhaustif.

Par ailleurs, dans la mesure où les mots de code sont de longueur fixe (ou bien si l'on utilisait des marqueurs de synchronisation), l'invention permet la création d'un code multiplexé permettant de décrire conjointement deux flux, dont un au moins bénéficie d'une synchronisation parfaite.

Annexe 1 - Formules

(E1) $\mathcal{A} = \{a_1, \ldots, a_i, \ldots, a_\Omega\}$

(E2) $\mu_i = P(a_i)$

(E3) $\tilde{\mu}_i = \frac{2^c}{2^c - \Omega_M} \mu_i$

(E4) $N_i = (2^c - \Omega_m) * \tilde{\mu}_i$

(E5) $\sum_{i \in \mathcal{A}} N_i = 2^c$

(E6) $c_{i,j} \rightleftarrows (s_i, q_j)$

(E7) $\Lambda = \prod_{t=1}^{K_H} n_t$

(E8) $K'_B = \lfloor log_2(\Lambda) \rfloor$

(E9) $\gamma = \sum_{r=1}^{K_H} b_{r+K_B-K'_B} 2^{r-1}$.

(E10) $u_T \text{bits} \rightleftarrows \begin{cases} v_{T,1} \text{ variables 2-valuée}, & e_{1,1}, e_{2,1}, \ldots, e_{v_{T,1},1} \\ v_{T,2} \text{ variables 3-valuée}, & e_{1,2}, e_{2,2}, \ldots, e_{v_{T,1},2} \\ \ldots & \\ v_{T,\nu} v_{T,\nu} \text{ variables } f_{\nu_i}\text{-valuée}, & e_{1,\nu}, e_{2,\nu}, \ldots, e_{v_{T,1},\nu} \end{cases}$

Annexe 1 - Formules

(E1) $\mathcal{A} = \{a_1, \ldots, a_i, \ldots, a_\Omega\}$

(E2) $\mu_i = P(a_i)$

(E3) $\tilde{\mu}_i = \frac{2^c}{2^c - \Omega_M} \mu_i$

(E4) $N_i = (2^c - \Omega_m) * \tilde{\mu}_i$

(E5) $\sum_{i \in \mathcal{A}} N_i = 2^c$

(E6) $c_{i,j} \rightleftarrows (s_i, q_j)$

(E7) $\Lambda = \prod_{t=1}^{K_H} n_t$

(E8) $K'_B = \lfloor log_2(\Lambda) \rfloor$

(E9) $\gamma = \sum_{r=1}^{K_H} b_{r+K_B-K'_B} 2^{r-1}.$

$$(E10) \qquad u_T \text{bits} \rightleftarrows \begin{cases} v_{T,1} \text{ variables 2-valuée}, & e_{1,1}, e_{2,1}, \ldots, e_{v_{T,1},1} \\ v_{T,2} \text{ variables 3-valuée}, & e_{1,2}, e_{2,2}, \ldots, e_{v_{T,1},2} \\ \ldots & \\ v_{T,\nu} v_{T,\nu} \text{ variables } f_{\nu_i}\text{-valuée}, & e_{1,\nu}, e_{2,\nu}, \ldots, e_{v_{T,1},\nu} \end{cases} \quad (1)$$

Annexe 2 - Algorithmes

A1

$\gamma' = \gamma$

Pour $t = 1$: K_H

  $q_t = \gamma'$ *modulo* n_t

  $\gamma' = \frac{\gamma' - q_t}{n_t}$

Fin pour

A2

$z = 0$

% Tant qu'il reste des variables f_j-valuées à obtenir

Tant que $sum(d_j) > 0$

  % Calcul du nombre de fois que la transformation $\mathcal{T}_z$ est utilisée

  $g_{\mathcal{T}_z} = floor(min(\frac{d_j}{v_{\mathcal{T}_z,j}}))$ où $v_{\mathcal{T}_z,j} \neq 0$

  % Calcul du nombre de variables f_j-valuée

  % qui n'ont pas été transformées par la transformation $\mathcal{T}_z$

  Pour chaque j *entre* 1 *et* ν

    $d_j = d_j - g_{\mathcal{T}_z} * v_{\mathcal{T}_z,j}$

    % Essaie la transformation suivante

    $z = z + 1$

A3

$e' = e$

Pour $j = 1$: ν

  Pour $r = 1$: $v_{\mathcal{T},j}$

    $e_{r,j} = e'$ *modulo* f_j

    $e' = \frac{e' - e_{r,j}}{f_j}$

  Fin pour

Fin pour

4 (15 pages) CABINET NETTER

Annexe 2 - Algorithmes

A1

$\gamma' = \gamma$

Pour $t = 1 : K_H$

  $q_t = \gamma'$ *modulo* n_t

  $\gamma' = \frac{\gamma' - q_t}{n_t}$

Fin pour

A2

$z = 0$

% Tant qu'il reste des variables f_j-valuées à obtenir

Tant que $sum(d_j) > 0$

  % Calcul du nombre de fois que la transformation $\mathcal{T}_z$ est utilisée

  $g_{\mathcal{T}_z} = floor(min(\frac{d_j}{v_{T_z,j}}))$ où $v_{T_z,j} \neq 0$

  % Calcul du nombre de variables f_j-valuée

  % qui n'ont pas été transformées par la transformation $\mathcal{T}_z$

  Pour chaque j *entre* 1 *et* ν

    $d_j = d_j - g_{\mathcal{T}_z} * v_{T_z,j}$

    % Essaie la transformation suivante

    $z = z + 1$

A3

$e' = e$

Pour $j = 1 : \nu$

  Pour $r = 1 : v_{T,j}$

    $e_{r,j} = e'$ *modulo* f_j

    $e' = \frac{e' - e_{r,j}}{f_j}$

  Fin pour

Fin pour

Revendications

1. Codeur de compression de données numériques, caractérisé en ce qu'il comprend:
- une entrée pour un premier flux de données (S_H), et un second flux de données (S_L),
- une table de codage, contenant une correspondance entre des symboles du premier flux de données et des mots de code, où, pour certains symboles, il existe plusieurs mots, dits redondants, correspondant au même symbole, et
- un module de traitement pour coder les symboles du premier flux de données d'après la table, en choisissant parmi les mots redondants, en fonction d'une partie au moins du second flux de données.

2. Codeur selon la revendication 1, caractérisé en ce que les mots de code sont de longueur fixe.

3. Codeur selon l'une des revendications 1 et 2, caractérisé en ce que le module de traitement comprend:
- une fonction de calcul de la capacité de multiplexage courante du premier flux de données (S_H), au vu de la table de codage, et
- une fonction d'extraction, dans le second flux de données (S_L), d'une partie multiplexée, déterminée d'après la capacité de multiplexage courante, pour être portée par lesdits mots redondants.

4. Codeur selon l'une des revendications précédentes, caractérisé en ce qu'il comprend une transformation d'un flux binaire en un flux de variable multi-valuée, en particulier en utilisant les transformations décrites dans la table 2.

5. Codeur selon l'une des revendications précédentes, caractérisé en ce que le second flux de données est préalablement encodé.

6. Codeur selon l'une des revendications précédentes, caractérisé en ce que le reste du second flux de données est concaténé aux données transmises.

Revendications

1. Codeur de compression de données numériques, caractérisé en ce qu'il comprend:
- une entrée pour un premier flux de données (S_H), et un second flux de données (S_L),
- une table de codage, contenant une correspondance entre des symboles du premier flux de données et des mots de code, où, pour certains symboles, il existe plusieurs mots, dits redondants, correspondant au même symbole, et
- un module de traitement pour coder les symboles du premier flux de données d'après la table, en choisissant parmi les mots redondants, en fonction d'une partie au moins du second flux de données.

2. Codeur selon la revendication 1, caractérisé en ce que les mots de code sont de longueur fixe.

3. Codeur selon l'une des revendications 1 et 2, caractérisé en ce que le module de traitement comprend:
- une fonction de calcul de la capacité de multiplexage courante du premier flux de données (S_H), au vu de la table de codage, et
- une fonction d'extraction, dans le second flux de données (S_L), d'une partie multiplexée, déterminée d'après la capacité de multiplexage courante, pour être portée par lesdits mots redondants.

4. Codeur selon l'une des revendications précédentes, caractérisé en ce qu'il comprend une transformation d'un flux binaire en un flux de variable multi-valuée, en particulier en utilisant les transformations décrites dans la table 2.

5. Codeur selon l'une des revendications précédentes, caractérisé en ce que le second flux de données est préalablement encodé.

6. Codeur selon l'une des revendications précédentes, caractérisé en ce que le reste du second flux de données est concaténé aux données transmises.

7. Décodeur, propre à effectuer les opérations inverses de celle du codeur de l'une des revendications précédentes.

8. Procédé de compression de données numériques, caractérisé par les étapes suivantes:

a. établir une table de codage, contenant une correspondance entre des symboles du premier flux de données et des mots de code, où, pour certains symboles, il existe plusieurs mots, dits redondants, correspondant au même symbole, et

b. coder les symboles d'un premier flux de données d'après la table, en choisissant parmi les mots redondants, en fonction d'une partie au moins d'un second flux de données.

9. Procédé selon la revendication 8, caractérisé par des sous-fonctions conformes à l'une des revendications 1 à 7.

10. Procédé de décompression de données numériques, caractérisé par les étapes réciproques de celles du procédé selon l'une des revendications 8 et 9.

α

7. Décodeur, propre à effectuer les opérations inverses de celle du codeur de l'une des revendications précédentes.

8. Procédé de compression de données numériques, caractérisé par les étapes suivantes:

a. établir une table de codage, contenant une correspondance entre des symboles du premier flux de données et des mots de code, où, pour certains symboles, il existe plusieurs mots, dits redondants, correspondant au même symbole, et

b. coder les symboles d'un premier flux de données d'après la table, en choisissant parmi les mots redondants, en fonction d'une partie au moins d'un second flux de données.

9. Procédé selon la revendication 8, caractérisé en ce que les mots de code sont de longueur fixe.

10. Procédé selon l'une des revendications 8 et 9, caractérisé en ce que l'étape b. comprend
- de calculer la capacité de multiplexage courante du premier flux de données (S_H), au vu de la table de codage, et
- d'extraire, dans le second flux de données (S_L), d'une partie multiplexée, déterminée d'après la capacité de multiplexage courante, pour être portée par lesdits mots redondants.

11. Procédé selon l'une des revendications 8 à 10, caractérisé en ce que l'étape b. comprend de transformer un flux binaire en un flux de variable multi-valuée, en particulier en utilisant les transformations décrites dans la table 2.

12. Procédé selon l'une des revendications 8 à 11, caractérisé en ce que le second flux de données est préalablement encodé.

13. Procédé selon l'une des revendications 8 à 12, caractérisé en ce que le reste du second flux de données est concaténé aux données transmises.

14. Procédé de décompression de données numériques, caractérisé par les étapes réciproques de celles du procédé selon l'une des revendications 8 à 13.

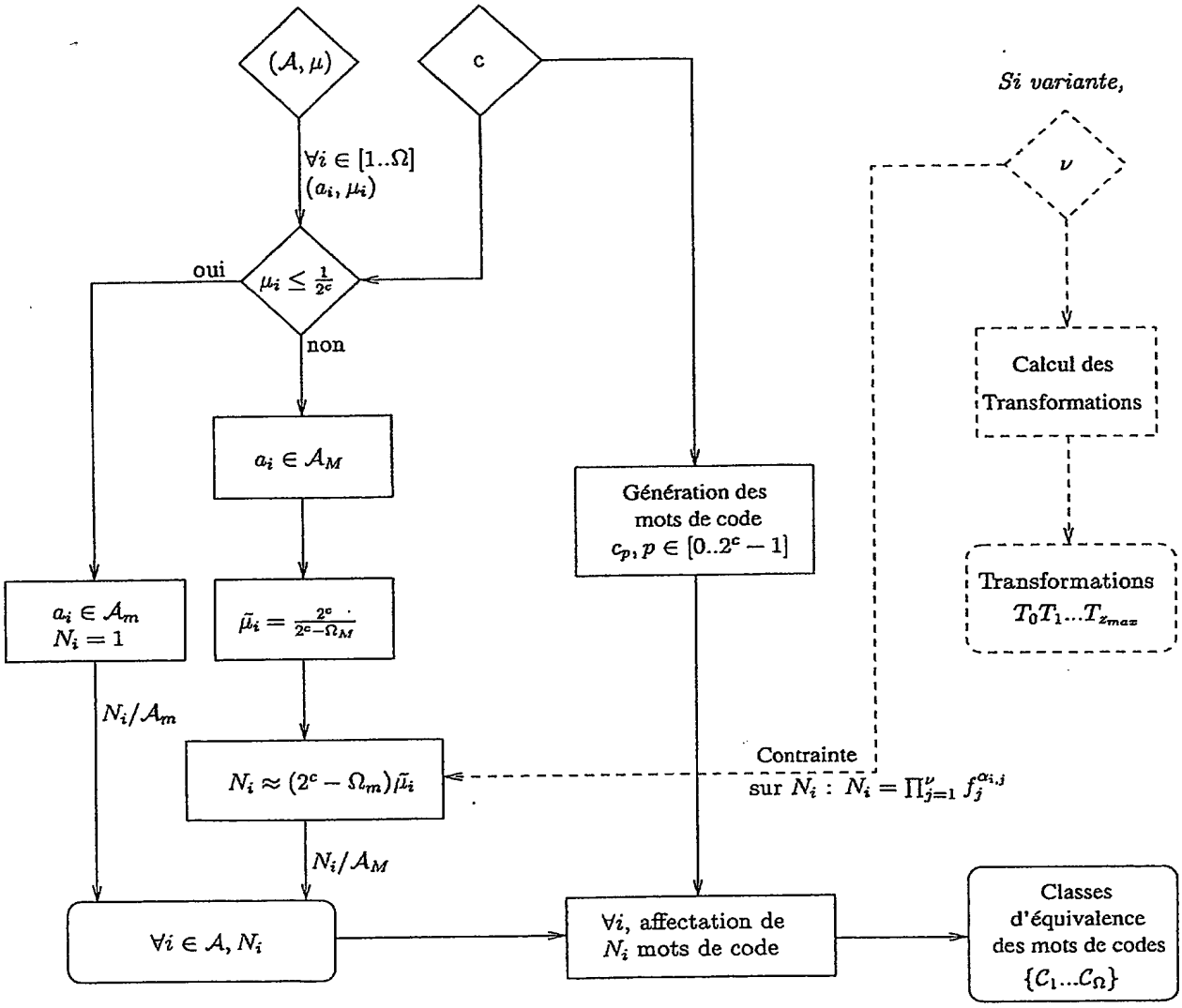
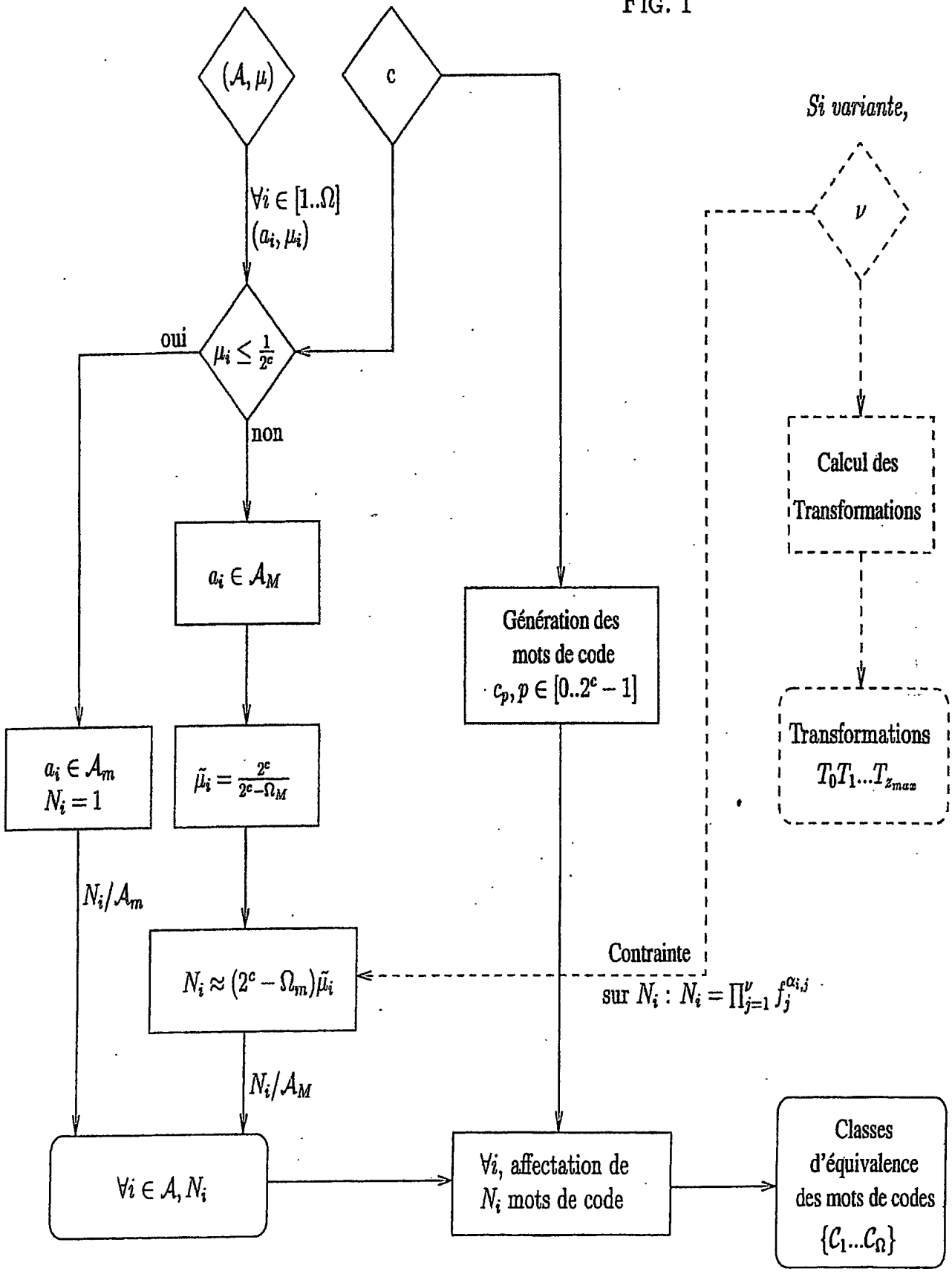


FIG. 1 – Procédé de création des codes

CABINET NETTER

FIG. 1


$(\mathcal{A}, \mu)$
c
$\forall i \in [1..\Omega]$
(a_i, μ_i)
$\mu_i \leq \frac{1}{2^c}$
oui
non
$a_i \in \mathcal{A}_M$
$a_i \in \mathcal{A}_m$
$N_i = 1$
$\tilde{\mu}_i = \frac{2^c}{2^c - \Omega_M}$
Génération des mots de code
$c_p, p \in [0..2^c - 1]$
$N_i / \mathcal{A}_m$
$N_i \approx (2^c - \Omega_m)\tilde{\mu}_i$
Contrainte
sur N_i : $N_i = \prod_{j=1}^{\nu} f_j^{\alpha_{i,j}}$
$N_i / \mathcal{A}_M$
$\forall i \in \mathcal{A}, N_i$
$\forall i$, affectation de N_i mots de code
Classes d'équivalence des mots de codes
$\{\mathcal{C}_1 ... \mathcal{C}_\Omega\}$
Si variante,
ν
Calcul des Transformations
Transformations
$T_0 T_1 ... T_{z_{max}}$

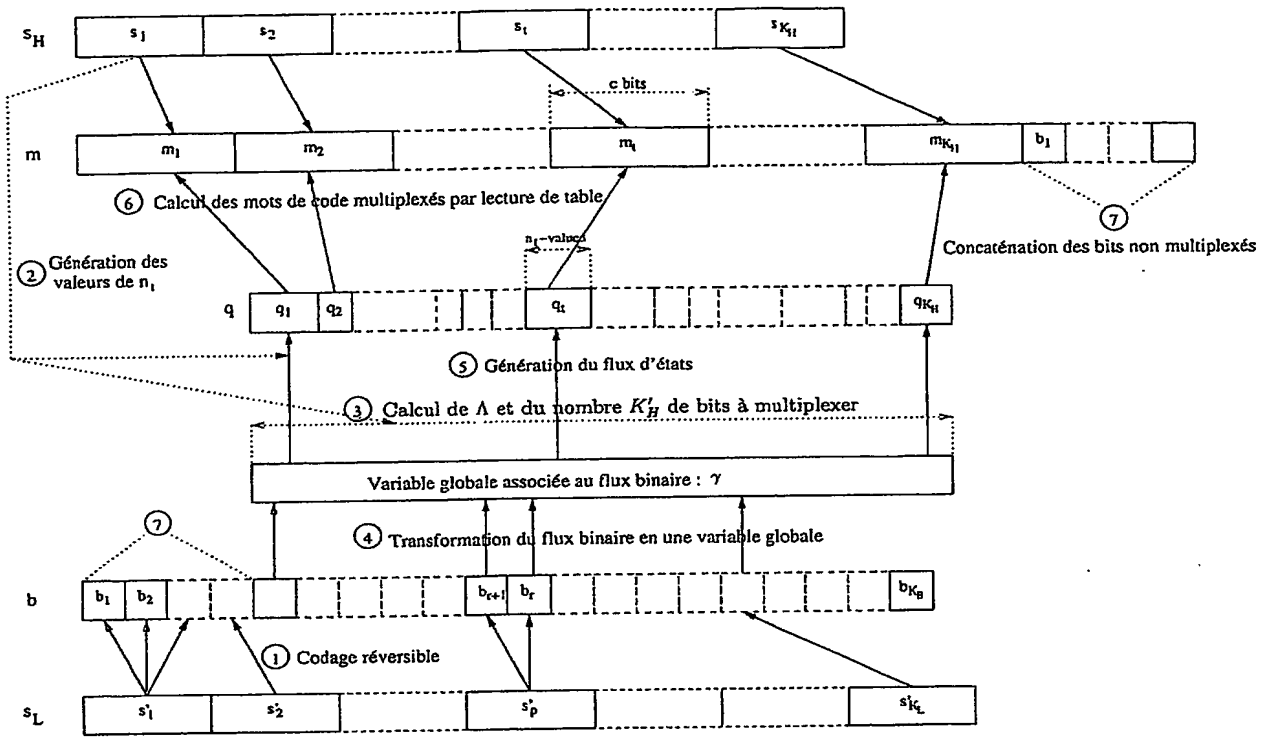


FIG. 2 – Vue générale du processus d'encodage

classe C_i	Mot de code $c_{i,q}$	symbole a_i	N_i=card(C_i)	probabilité μ_i	état q
C_1	0000	a	6	0.43	0
	0001				1
	0010				2
	0011				3
	0100				4
	0101				5
C_2	0110	b	5	0.30	0
	0111				1
	1000				2
	1001				3
	1010				4
C_3	1011	c	4	0.25	0
	1100				1
	1101				2
	1110				3
C_4	1111	d	1	0.02	0

TAB. 1 – Un exemple de code multiplexé ($c = 4$). L'alphabet ne contient ici que 4 éléments, a, b, c, d.

CABINET NETTER

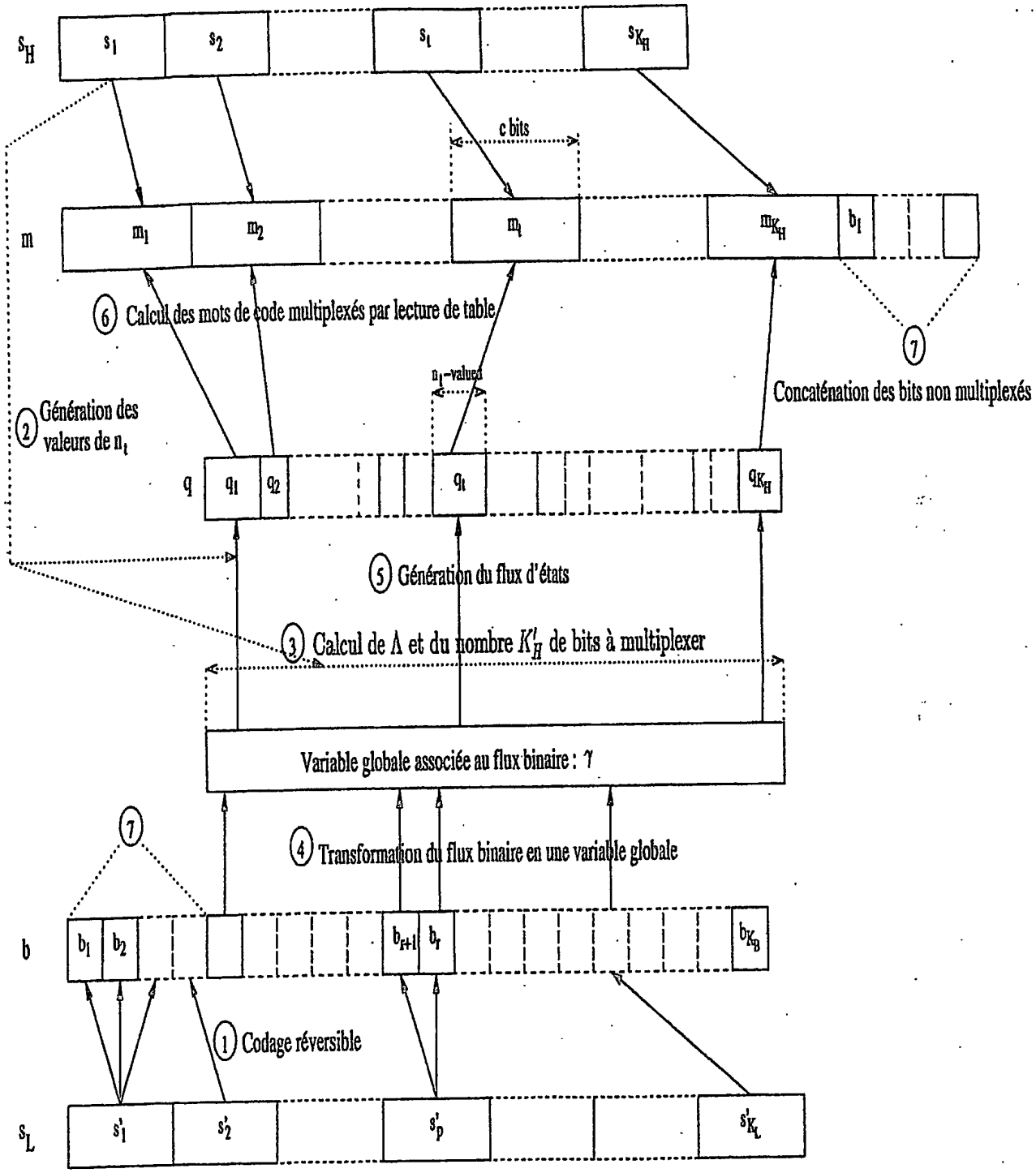

s_H
s_1
s_2
s_t
s_{K_H}
c bits
m
m_1
m_2
m_t
m_{K_H}
b_1
6 Calcul des mots de code multiplexés par lecture de table
7
Concaténation des bits non multiplexés
n_t-valued
2 Génération des valeurs de n_t
q
q_1
q_2
q_t
q_{K_H}
5 Génération du flux d'états
3 Calcul de A et du nombre K'_H de bits à multiplexer
Variable globale associée au flux binaire : γ
7
4 Transformation du flux binaire en une variable globale
b
b_1
b_2
b_{r+1}
b_r
b_{K_B}
1 Codage réversible
s_L
s'_1
s'_2
s'_p
s'_{K_L}

FIG. 2

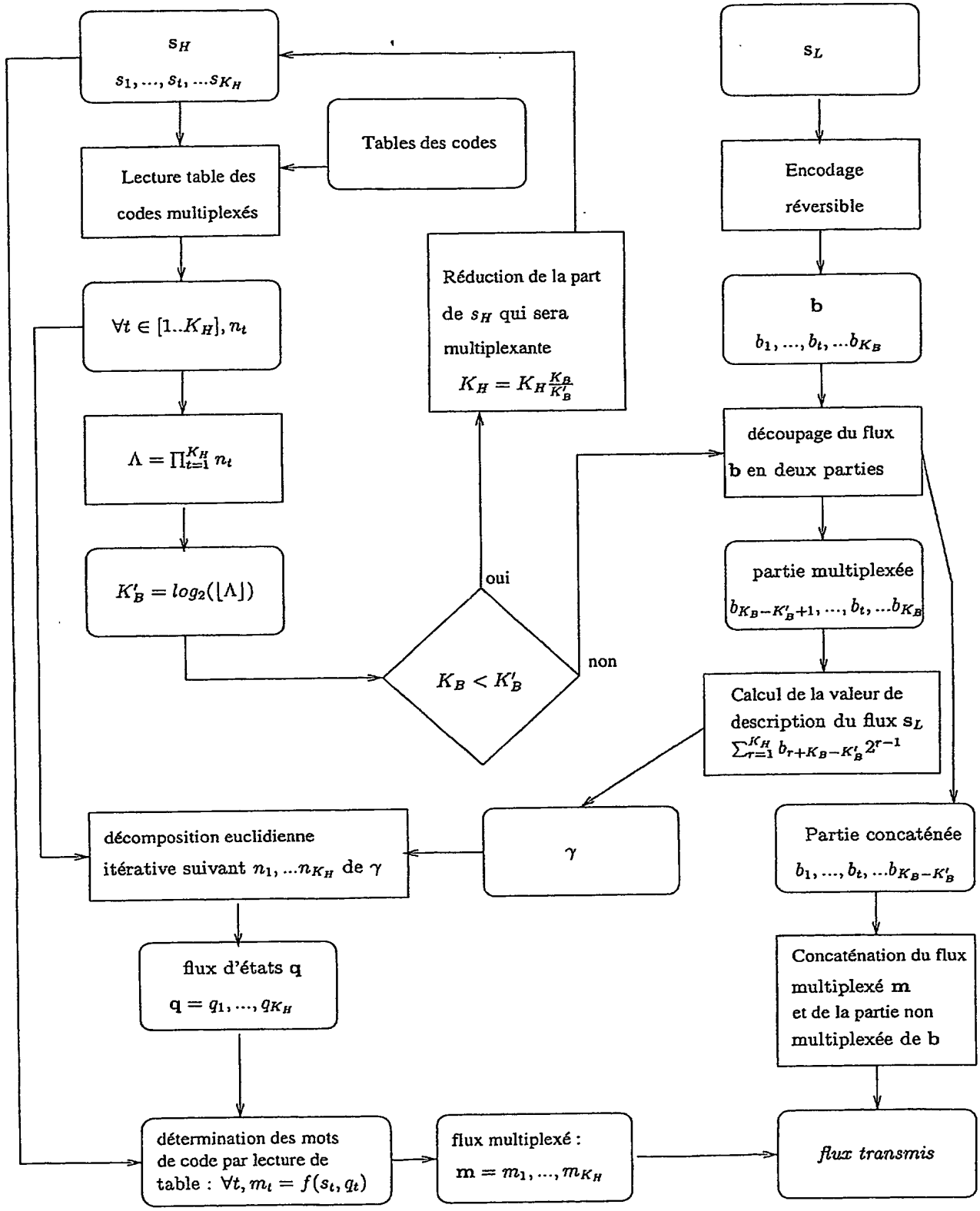


FIG. 3 – Procédé d'encodage

CABINET NETTER

classe $\mathcal{C}_i$	Mot de code $c_{i,q}$	symbole a_i	N_i=card($\mathcal{C}_i$)	probabilité μ_i	état q
$\mathcal{C}_1$	0000	a	6	0.43	0
	0001				1
	0010				2
	0011				3
	0100				4
	0101				5
$\mathcal{C}_2$	0110	b	5	0.30	0
	0111				1
	1000				2
	1001				3
	1010				4
$\mathcal{C}_3$	1011	c	4	0.25	0
	1100				1
	1101				2
	1110				3
$\mathcal{C}_4$	1111	d	1	0.02	0

TAB. 1

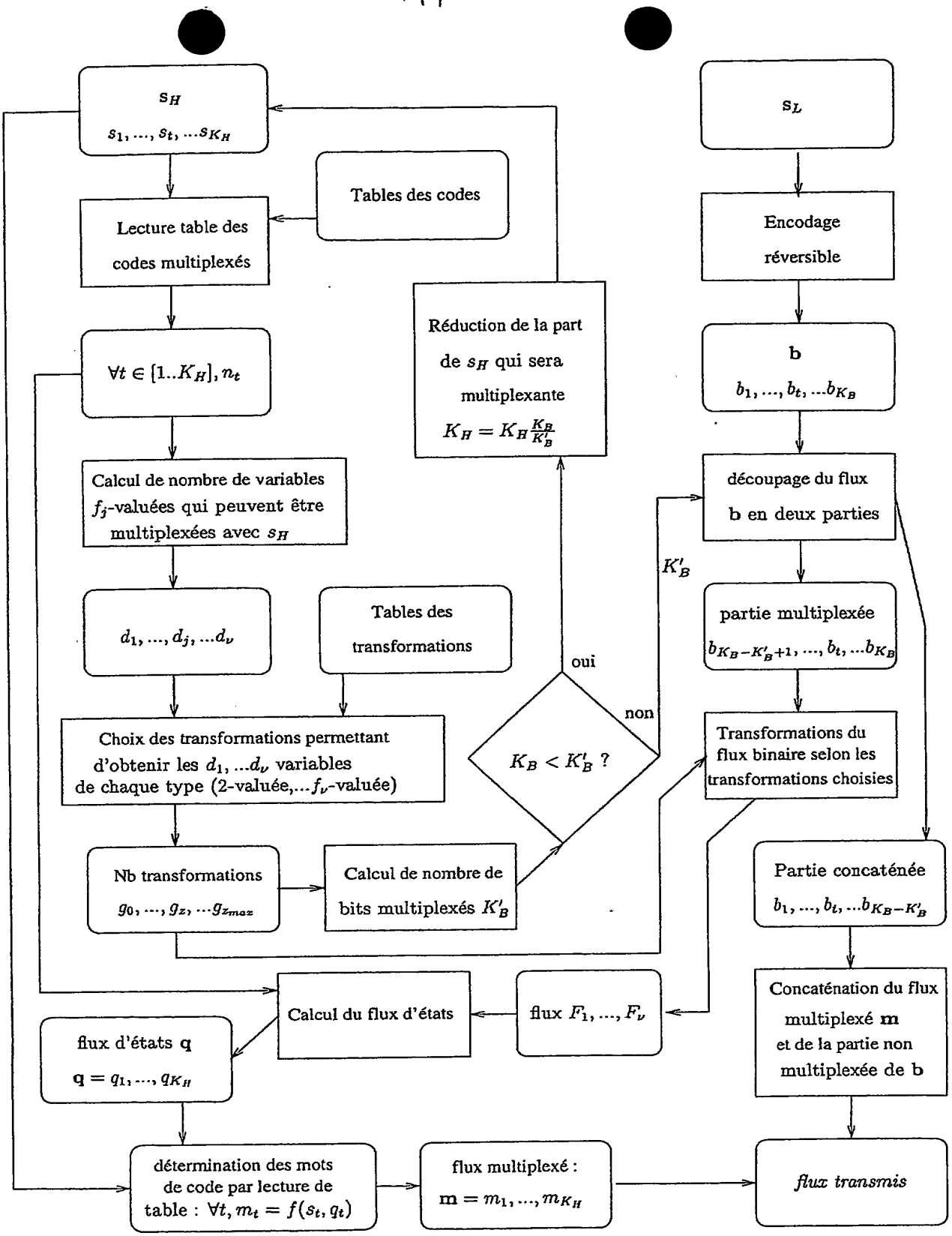


FIG. 4 – Procédé d'encodage : variante

CABINET NETTER

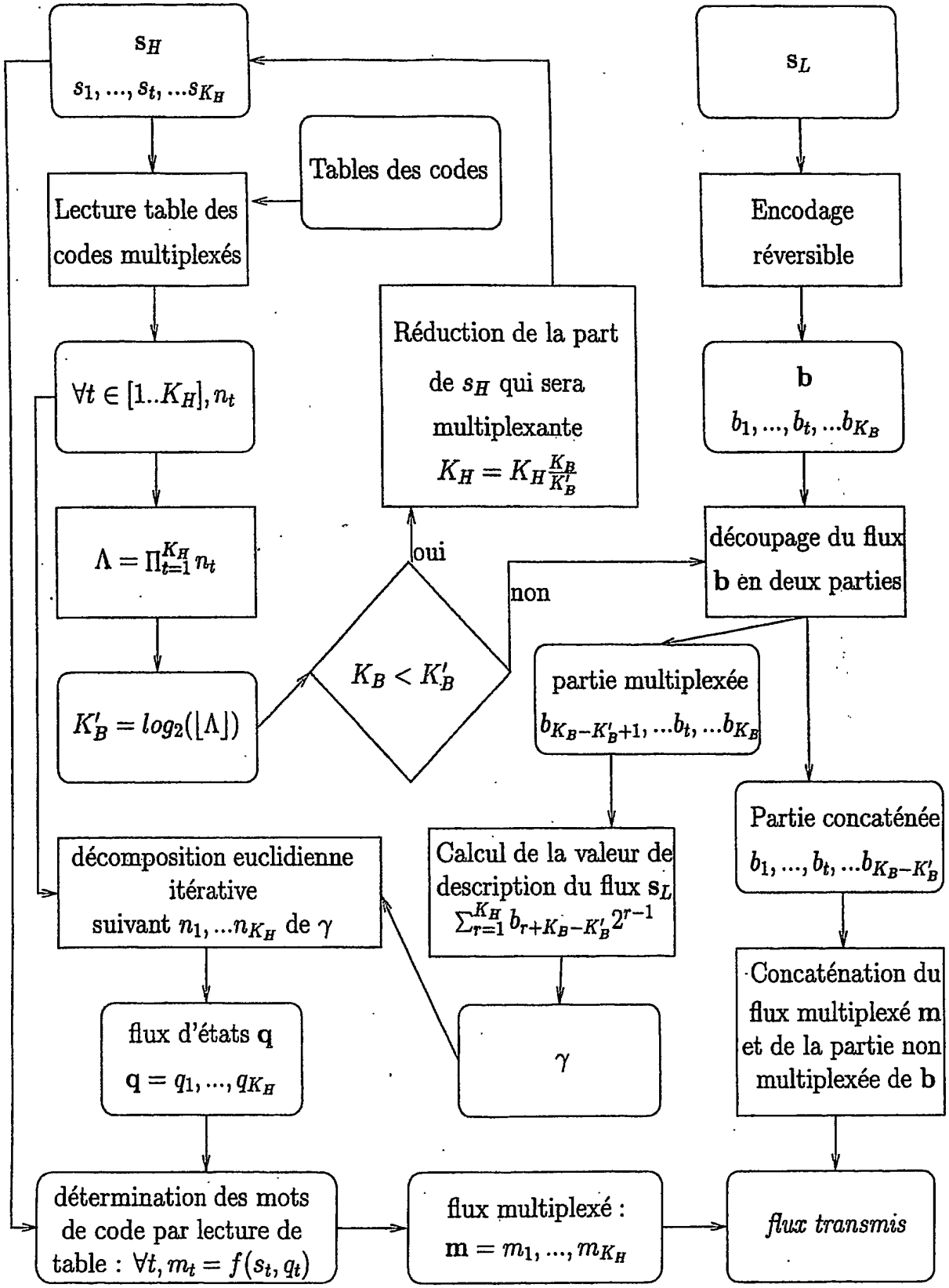

s_H
$s_1, ..., s_t, ...s_{K_H}$
s_L
Tables des codes
Lecture table des codes multiplexés
Encodage réversible
Réduction de la part de s_H qui sera multiplexante
$K_H = K_H \frac{K_B}{K'_B}$
$\forall t \in [1..K_H], n_t$
b
$b_1, ..., b_t, ...b_{K_B}$
$\Lambda = \Pi_{t=1}^{K_H} n_t$
découpage du flux b en deux parties
oui
non
$K_B < K'_B$
$K'_B = log_2(\lfloor \Lambda \rfloor)$
partie multiplexée
$b_{K_B-K'_B+1}, ...b_t, ...b_{K_B}$
Partie concaténée
$b_1, ..., b_t, ...b_{K_B-K'_B}$
décomposition euclidienne itérative suivant $n_1, ...n_{K_H}$ de γ
Calcul de la valeur de description du flux s_L
$\sum_{r=1}^{K_H} b_{r+K_B-K'_B} 2^{r-1}$
Concaténation du flux multiplexé m et de la partie non multiplexée de b
flux d'états q
$q = q_1, ..., q_{K_H}$
γ
détermination des mots de code par lecture de table : $\forall t, m_t = f(s_t, q_t)$
flux multiplexé :
$m = m_1, ..., m_{K_H}$
flux transmis

FIG. 3

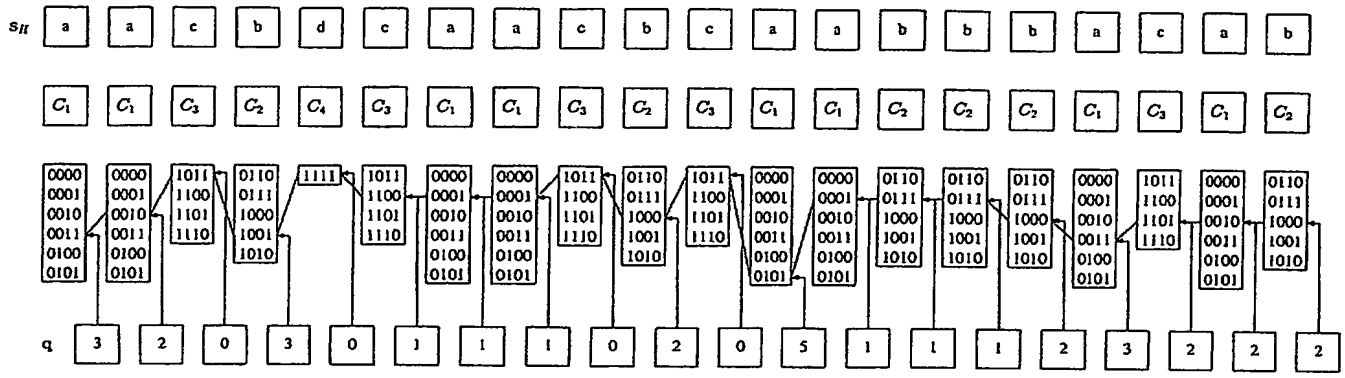
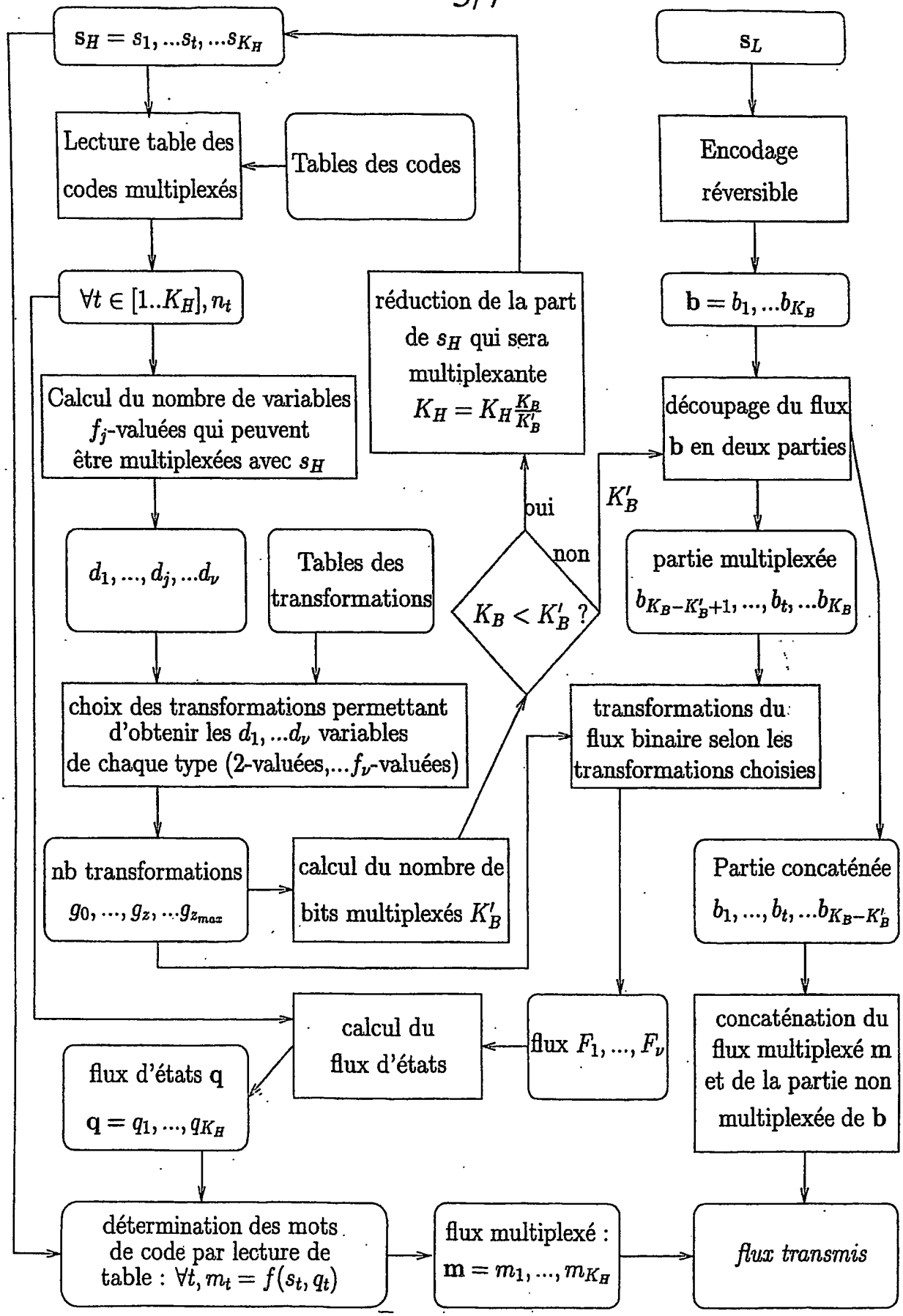


FIG. 5 – Exemple de création d'une capacité de stockage par l'attribution de plusieurs mots de codes à un symbole. Un flux de données $\mathbf{q}$ peut être conjointement stocké.

$\mathcal{T}_z$ identificateur	$u_{\mathcal{T}_z}$	$v_{\mathcal{T}_z,1}$	$v_{\mathcal{T}_z,2}$	$v_{\mathcal{T}_z,3}$	$o_{\mathcal{T}_z}$
$\mathcal{T}_0$	1	1	0	0	0.0000
$\mathcal{T}_1$	15	0	8	1	0.0001
$\mathcal{T}_2$	21	0	3	7	0.0004
$\mathcal{T}_3$	19	0	12	0	0.0010
$\mathcal{T}_4$	25	0	7	6	0.0011
$\mathcal{T}_5$	24	0	2	9	0.0028
$\mathcal{T}_6$	14	0	3	4	0.0030
$\mathcal{T}_7$	18	0	7	3	0.0034
$\mathcal{T}_8$	27	0	1	11	0.0047
$\mathcal{T}_9$	17	0	2	6	0.0060
$\mathcal{T}_{10}$	30	0	0	13	0.0062
$\mathcal{T}_{11}$	20	0	1	8	0.0080
$\mathcal{T}_{12}$	11	0	7	0	0.0086
$\mathcal{T}_{13}$	23	0	0	10	0.0095
$\mathcal{T}_{14}$	6	0	1	2	0.0381
$\mathcal{T}_{15}$	3	0	2	0	0.0566
$\mathcal{T}_{16}$	2	0	0	1	0.1610
$\mathcal{T}_{z_{max}} = \mathcal{T}_{17}$	1	0	1	0	0.5850

TAB. 2 – Transformations utilisées pour $f_\nu = 5$

CABINET NETTER


$\mathbf{s}_H = s_1, ...s_t, ...s_{K_H}$
$\mathbf{s}_L$
Lecture table des codes multiplexés
Tables des codes
Encodage réversible
$\forall t \in [1..K_H], n_t$
réduction de la part de s_H qui sera multiplexante $K_H = K_H \frac{K_B}{K'_B}$
$\mathbf{b} = b_1, ...b_{K_B}$
Calcul du nombre de variables f_j-valuées qui peuvent être multiplexées avec s_H
découpage du flux b en deux parties
K'_B
oui
non
$d_1, ..., d_j, ...d_\nu$
Tables des transformations
$K_B < K'_B$?
partie multiplexée $b_{K_B-K'_B+1}, ..., b_t, ...b_{K_B}$
choix des transformations permettant d'obtenir les $d_1, ...d_\nu$ variables de chaque type (2-valuées,...f_ν-valuées)
transformations du flux binaire selon les transformations choisies
nb transformations $g_0, ..., g_z, ...g_{z_{max}}$
calcul du nombre de bits multiplexés K'_B
Partie concaténée $b_1, ..., b_t, ...b_{K_B-K'_B}$
calcul du flux d'états
flux $F_1, ..., F_\nu$
concaténation du flux multiplexé m et de la partie non multiplexée de b
flux d'états q $\mathbf{q} = q_1, ..., q_{K_H}$
détermination des mots de code par lecture de table : $\forall t, m_t = f(s_t, q_t)$
flux multiplexé : $\mathbf{m} = m_1, ..., m_{K_H}$
flux transmis

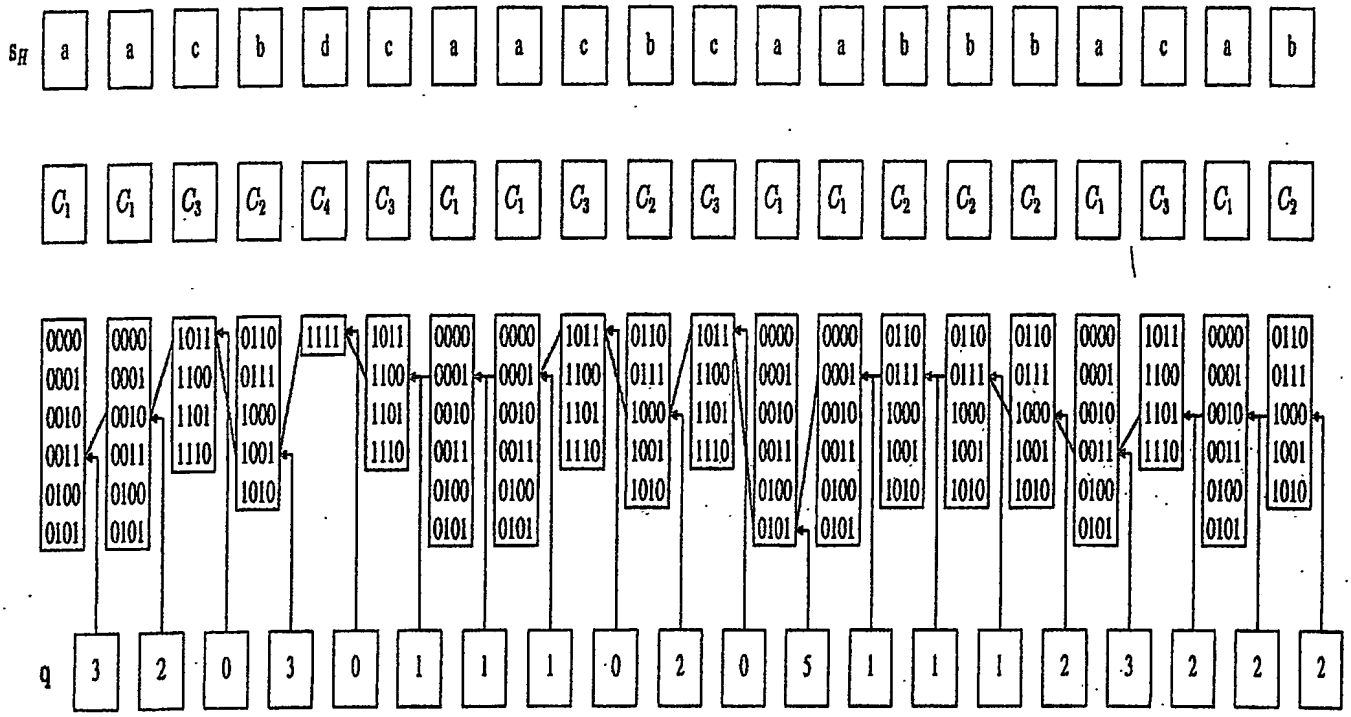

s_H
a a c b d c a a c b c a a b b b a c a b
C_1 C_1 C_3 C_2 C_4 C_3 C_1 C_1 C_3 C_2 C_3 C_1 C_1 C_2 C_2 C_2 C_1 C_3 C_1 C_2
q
3 2 0 3 0 1 1 1 0 2 0 5 1 1 1 2 3 2 2 2

FIG. 5

$\mathcal{T}_z$ identificateur	$u_{\mathcal{T}_z}$	$v_{\mathcal{T}_z,1}$	$v_{\mathcal{T}_z,2}$	$v_{\mathcal{T}_z,3}$	$o_{\mathcal{T}_z}$
$\mathcal{T}_0$	1	1	0	0	0.0000
$\mathcal{T}_1$	15	0	8	1	0.0001
$\mathcal{T}_2$	21	0	3	7	0.0004
$\mathcal{T}_3$	19	0	12	0	0.0010
$\mathcal{T}_4$	25	0	7	6	0.0011
$\mathcal{T}_5$	24	0	2	9	0.0028
$\mathcal{T}_6$	14	0	3	4	0.0030
$\mathcal{T}_7$	18	0	7	3	0.0034
$\mathcal{T}_8$	27	0	1	11	0.0047
$\mathcal{T}_9$	17	0	2	6	0.0060
$\mathcal{T}_{10}$	30	0	0	13	0.0062
$\mathcal{T}_{11}$	20	0	1	8	0.0080
$\mathcal{T}_{12}$	11	0	7	0	0.0086
$\mathcal{T}_{13}$	23	0	0	10	0.0095
$\mathcal{T}_{14}$	6	0	1	2	0.0381
$\mathcal{T}_{15}$	3	0	2	0	0.0566
$\mathcal{T}_{16}$	2	0	0	1	0.1610
$\mathcal{T}_{z_{max}} = \mathcal{T}_{17}$	1	0	1	0	0.5850

TAB. 2

This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☑ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☑ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.